

# Piecewise-Deterministic Optimal Path-Planning

Z. Shen\* and A. Vladimirovsky\*

Center for Applied Mathematics and Department of Mathematics  
Cornell University, Ithaca, NY 14853

## Abstract

We consider piecewise-deterministic optimal control problems in which the environment randomly switches among several deterministic modes, and the goal is to optimize the expected cost up to the termination while taking the likelihood of future mode-switches into account. The dynamic programming approach yields a weakly-coupled system of Hamilton-Jacobi-Bellman PDEs satisfied by the value functions of individual modes. We derive and implement semi-Lagrangian and Eulerian numerical schemes for that system. We further recover simpler, “asymptotically optimal” controls and test their performance in non-asymptotic regimes. Our approach is illustrated on a simple anisotropic path-planning problem: the time-optimal control for a boat affected by randomly switching winds.

## 1 Introduction

Dynamic programming is a popular approach for solving continuous optimal control problems with a moderate number of degrees of freedom  $d$ . The value function is defined on a  $d$ -dimensional domain to encode the minimum-cost-till-termination for each initial configuration. Bellman’s optimality principle allows to recover the value function by solving the corresponding Hamilton-Jacobi-Bellman (HJB) partial differential equation [5]. If the controlled process is deterministic, that PDE involves only the gradient of the value function. On the other hand, the most common setting for the stochastic optimal control is based on the assumption that the deterministic dynamics is perturbed by the additive Brownian motion, yielding second partial derivatives in the equation [25].

Piecewise-deterministic controlled systems [20, 40] occupy an intermediate niche and are useful for encoding non-diffusive stochastic perturbations. In this framework, the assumption is that both the cost and dynamics remain deterministic except for occasional transitions to different deterministic modes. The value functions of different modes satisfy a weakly-coupled system of first-order HJB equations, which is more expensive computationally. Nevertheless, this formulation is extensively used to model the optimal control of manufacturing processes (e.g., [1, 9, 34, 36]), production/maintenance planning [12], economic growth & global climate change modeling [28], as well as multi-generational games [27]. More generally, they can be used to optimize the expected performance of any *stochastic switching system*, provided the total number of possible environment configurations is moderate, the environment changes are random in nature, and the planner becomes aware of the exact change as soon as it happens. We believe that they can be particularly useful in path-planning problems, where describing stochastic perturbations by additive Brownian motion is often unsatisfactory [41]. Another related problem is the deterministic optimal control of randomly-terminated processes recently considered in [3].

For a concrete example, consider a generalization of the classical “Zermelo’s navigation problem” [43]. We will assume that, in the absence of winds, a rowboat can move with a (location-dependent)

---

\*SUPPORTED IN PART BY THE NATIONAL SCIENCE FOUNDATION GRANT DMS-1016150.

speed  $s(\mathbf{x})$ . In this case, the total time to the target  $Q$  can be minimized by using the gradient descent in the value function  $v$ , defined by the Eikonal equation  $\|\nabla v\|s(\mathbf{x}) = 1$  with  $v = 0$  on  $Q$ . If we now add wind which moves that boat with velocity  $\mathbf{w}(\mathbf{x})$ , the boat dynamics becomes anisotropic. For a unit vector  $\mathbf{a}$  encoding the direction in which we are currently rowing, the resulting boat velocity is  $\mathbf{f}(\mathbf{x}, \mathbf{a}) = s(\mathbf{x})\mathbf{a} + \mathbf{w}(\mathbf{x})$ , and the corresponding HJB equation is

$$\|\nabla v\|s(\mathbf{x}) - \mathbf{w}(\mathbf{x}) \cdot \nabla v = 1.$$

This equation can be solved by a variety of fast numerical methods for static HJB developed in the last 15 years [38, 39, 2, 32, 11, 30, 35, 10, 29, 26].

But what if we have several different “wind maps” ( $\mathbf{w}_i(\mathbf{x})$ ,  $i = 1, \dots, n$ ) commonly observed on this reservoir? It is easy to define  $\mathbf{f}_i$  for each  $\mathbf{w}_i$  and then solve for each corresponding  $v_i$  separately. The relevant wind map  $\mathbf{w}_i$  is selected when the boat starts moving, and the path is planned based on  $v_i$ . However, the wind direction might change before we reach the target. An intuitive strategy is to react by switching to a new path (based on the new wind map) as soon as this change occurs. But if we have statistics on how often such switches happen, this can be used to improve our path planning. Suppose  $\lambda_{ij} \geq 0$  characterizes the rate of switching from  $\mathbf{w}_i$  to  $\mathbf{w}_j$ . The new value function  $u_i$  is defined as the minimal expected time to target if we start from  $\mathbf{x}$  when the wind map  $\mathbf{w}_i$  is in effect. It is easy to show that such value functions must satisfy a *weakly-coupled* system:

$$\|\nabla u_i\|s(\mathbf{x}) - \mathbf{w}_i(\mathbf{x}) \cdot \nabla u_i = 1 - \sum_{j=1}^n \lambda_{ij}(u_i - u_j), \quad i = 1, \dots, n. \quad (1.1)$$

More general piecewise-deterministic systems are described in detail in section 2 and the numerical methods for them are discussed in section 4.

Since solving a coupled system is more expensive, the natural question is whether path-planning based on  $v_i$ ’s would result in a significantly larger expected time of arrival. We note that the controls based on  $v_i$ ’s can be viewed as asymptotically optimal if all transition rates tend to zero. A different approach to reduce the computational cost is based on a *single* value function  $u^\infty$  describing the optimal behavior of a controller who does not keep track of the current mode of dynamics [41, 13]. In section 3, we explain why this strategy can be also viewed as “asymptotically optimal” when all rates of transition tend to infinity. We then test the performance of both asymptotically optimal strategies with finite, positive  $\lambda_{ij}$  values. In section 5, we show that both of them result in a noticeable performance degradation, with the strategy based on  $u^\infty$  yielding collisions with obstacles in a significant portion of Monte Carlo simulations.

Our example used for this benchmarking is particularly simple: a single rectangular obstacle and only two space-homogeneous wind-maps. Yet, we show that the resulting dynamics based on the correct  $u_i$ ’s has several subtle features. E.g., it is well-known that in deterministic path planning, optimal trajectories are piecewise straight lines if the velocity is the same throughout the domain. Our tests in section 5 show that this is no longer the case in a piecewise-deterministic setting. In section 6 we consider a different limiting case, with the number of modes/wind directions tending to infinity, and show the connection to degenerate elliptic HJB PDEs on  $\mathbb{R}^{d+1}$ . We conclude by discussing several directions for future work in section 7.

## 2 General Problem Statement

Consider the following feedback control process in a bounded domain  $\Omega \subset \mathbb{R}^d$ :

$$\begin{cases} \mathbf{y}'(t) &= \mathbf{f}_{m(t)}(\mathbf{y}(t), \alpha(\mathbf{y}(t), m(t))), \\ \mathbf{y}(0) &= \mathbf{x}, \\ m(0) &= i. \end{cases} \quad (2.1)$$

The full state of the process is specified by its continuous component  $\mathbf{y}(t)$  and the current “mode” of the dynamics  $m(t) \in \mathcal{M} = \{1, \dots, n\}$ . The continuous component changes according to a differential equation, while the mode  $m(t)$  randomly switches as a continuous-time Markov process on  $\mathcal{M}$ . The right hand side in the ODE for the continuous component depends on the current state  $(\mathbf{y}, m)$  and the current control  $\alpha(\mathbf{y}, m)$  chosen from a compact set of control values  $A$ . We note that  $\mathbf{y}(t)$  is a random variable which also depends on the initial conditions  $(\mathbf{x}, i)$ , and the control  $\alpha \in \mathcal{A}$ , but we omit these dependencies from the notation for the sake of readability.

As usual, the goal is to find an *optimal feedback control function* from the admissible set  $\mathcal{A} = \{ \text{measurable } \alpha : \Omega \times \mathcal{M} \mapsto A \}$ , but before we specify the optimization criteria, we need to describe the mode-switching process  $m(t)$ . It can be characterized by the transition rate matrix  $\Lambda = (\lambda_{ij})$ , where  $\lambda_{ij}$  is the transition rate from mode  $i$  to mode  $j$ :

$$\lim_{\tau \rightarrow 0} \frac{Pr(m(\tau) = j | m(0) = i) - \delta_{ij}}{\tau} = \lambda_{ij}, \quad \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad (2.2)$$

It is easy to verify that the transition rate matrix should satisfy

$$\begin{aligned} \lambda_{ij} &\geq 0, & \forall j \neq i; \\ \lambda_{ii} &= - \sum_{j \neq i} \lambda_{ij}, & \forall i. \end{aligned} \quad (2.3)$$

We also recall several standard properties of continuous-time Markov processes useful in our context:

- At any time  $t$ , conditional on  $m(t) = i$ , the probability that the system stays in current mode for at least time  $s$  is  $\exp(\lambda_{ii}s) = \exp(-\sum_{j \neq i} \lambda_{ij}s)$ .
- If we define  $p_{ij}(t) = Pr(m(t) = j | m(0) = i)$ , the evolution of the probability matrix  $P(t) = (p_{ij}(t))$  is then described by

$$\frac{d}{dt} P(t) = \Lambda P(t) = P(t) \Lambda, \quad P(0) = I.$$

Therefore,  $P(t) = \exp(\Lambda t)$ .

Given a closed “target set”  $Q \subset \bar{\Omega}$ , we define the *stopping time*  $T = \min_{t \geq 0} \{t \mid \mathbf{y}(t) \in Q\}$ . The total (random) cost of starting from  $(\mathbf{x}, i)$  and using the control  $\alpha(\cdot)$  is defined as

$$J(\mathbf{x}, i, \alpha) = \int_0^T C_{m(t)}(\mathbf{y}(t), \alpha(\mathbf{y}(t), m(t))) dt + q(\mathbf{y}(T)), \quad (2.4)$$

where  $C$  is the positive *running cost* defined on  $(\Omega \setminus Q) \times \mathcal{M} \times A$  and  $q$  is the *terminal cost* defined on  $Q \cup \partial\Omega$ . To constrain the dynamics to  $\Omega$ , we impose a prohibitive terminal cost ( $q = +\infty$ ) on  $\partial\Omega \setminus Q$ . We can now define the *value function* as the minimum expected cost starting from  $\mathbf{x}$  with initial mode  $i$ :

$$u(\mathbf{x}, i) = \inf_{\alpha \in \mathcal{A}} \mathbb{E}[J(\mathbf{x}, i, \alpha)] \quad (2.5)$$

Throughout this paper we will use  $u_i(\mathbf{x})$  and  $u(\mathbf{x}, i)$  notations interchangeably.

A standard formal argument can be used to derive the Hamilton-Jacobi-Bellman (HJB) equation for  $u$ . The optimality of  $\alpha \in \mathcal{A}$  yields

$$u(\mathbf{x}, i) = \min_{\alpha \in \mathcal{A}} \left\{ \mathbb{E} \left[ \int_0^\tau C_{m(t)}(\mathbf{y}(t), \alpha(\mathbf{y}(t), m(t))) dt \right] + \mathbb{E}[u(\mathbf{y}(\tau), m(\tau))] \right\} \quad (2.6)$$

for  $\tau$  less than the stopping time. As  $\tau \rightarrow 0$ , we have

$$u(\mathbf{x}, i) = \min_{\mathbf{a} \in A} \left\{ \tau C_i(\mathbf{x}, \mathbf{a}) + \sum_{j=1}^n p_{ij}(\tau) u(\tilde{\mathbf{x}}_{\mathbf{a}}, j) \right\} + o(\tau), \quad (2.7)$$

where  $\tilde{\mathbf{x}}_{\mathbf{a}} = \mathbf{x} + \tau \mathbf{f}_i(\mathbf{x}, \mathbf{a})$ .

The first-order approximations of  $p_{ij}(\tau)$  and  $u(\tilde{\mathbf{x}}_{\mathbf{a}}, i)$  are:

$$\begin{aligned} p_{ij}(\tau) &= 1 - \exp(-\lambda_{ij}\tau) + o(\tau) = \lambda_{ij}\tau + o(\tau), & \text{if } j \neq i, \\ p_{ii}(\tau) &= 1 - \sum_{j \neq i} p_{ij}(\tau) = 1 - \sum_{j \neq i} \lambda_{ij}\tau + o(\tau), \\ u_i(\tilde{\mathbf{x}}_{\mathbf{a}}) &= u_i(\mathbf{x}) + \tau \mathbf{f}_i(\mathbf{x}, \mathbf{a}) \cdot \nabla u_i(\mathbf{x}) + o(\tau). \end{aligned}$$

Using these approximations in (2.7) we obtain

$$u_i(\mathbf{x}) = \min_{\mathbf{a} \in A} \left\{ \tau C_i(\mathbf{x}, \mathbf{a}) + \sum_{j \neq i} \lambda_{ij}\tau u_j(\mathbf{x}) + \left( u_i(\mathbf{x}) + \tau \mathbf{f}_i(\mathbf{x}, \mathbf{a}) \cdot \nabla u_i(\mathbf{x}) - \sum_{j \neq i} \lambda_{ij}\tau u_i(\mathbf{x}) \right) \right\} + o(\tau). \quad (2.8)$$

Simplifying and letting  $\tau \rightarrow 0$ , we obtain the HJB equation:

$$\min_{\mathbf{a} \in A} \{ \nabla u_i(\mathbf{x}) \cdot \mathbf{f}_i(\mathbf{x}, \mathbf{a}) + C_i(\mathbf{x}, \mathbf{a}) \} - \sum_{j \neq i} \lambda_{ij} (u_i(\mathbf{x}) - u_j(\mathbf{x})) = 0, \quad (2.9)$$

with  $u_i(\mathbf{x}) = q(\mathbf{x})$  on  $Q \cup \partial\Omega$ . Since this has to hold for every  $i \in \mathcal{M}$ , it is more natural to think of this as a system of  $n$  HJB PDEs. We note that this system is *weakly coupled* as long as  $\lambda$ 's are non-zero.

The above derivation is formal since these PDEs typically don't admit classical solutions even for  $n = 1$ . To obtain a derivation for the most general case, we need the concept of viscosity solution, which was first introduced in [17]. The viscosity solution of similar stochastic hybrid control problems is discussed in [8].

### 3 Mode Differences and Asymptotic Approximations

The coupling in PDEs (2.9) presents significant challenges for the numerical computation of the value function and implementation of optimal controls. The mode-switching results in the interdependence of value functions for specific modes, increasing the number of iterations needed to solve the discretization of these PDEs. To alleviate these difficulties, it can be useful to consider simplified/asymptotic versions of the above system. If all rates of switching approach zero, the system becomes uncoupled (with many efficient numerical methods available). Alternatively, we also consider the infinite-transition-rate case, in which the switches happen so often that it is not necessary (or is no longer possible in practice) to keep track of the current mode.

Both of these asymptotic approximations can be used instead of the correct value function  $u$  (based on the actual switching rates matrix  $\Lambda$ ) to simplify the computation of optimal controls. The performance implications of this approach are explored in the simulations of section 5.

### 3.1 Uncoupled Planning

If there is no switching between different modes (i.e.,  $\Lambda = 0$ ), the corresponding value function, defined as  $u_i^0(\mathbf{x})$ , must satisfy the system of HJBs

$$\min_{\mathbf{a} \in A} \{ \nabla u_i^0(\mathbf{x}) \cdot \mathbf{f}_i(\mathbf{x}, \mathbf{a}) + C_i(\mathbf{x}, \mathbf{a}) \} = 0. \quad (3.1)$$

Since this system is *uncoupled*,  $u^0$  can be computed for each mode  $i$  separately<sup>1</sup>. This is simply a collection of  $n$  standard deterministic optimal control problems, for which many efficient numerical methods are already available. Methods based on Fast Sweeping [11, 42, 35, 30], generalizations of Fast Marching [37, 38, 39, 2, 32], hybrid two-scale methods [14, 15], or other label-correcting-type techniques [10, 4, 29, 26] might be advantageous depending on what is known about the running cost and the dynamics. Crucially, many of these take advantage of the direction of characteristics, which are typically quite different for each mode  $i$ . This is not a problem in the uncoupled case, but makes such techniques much less efficient when we are interested in a general  $\Lambda$ .

### 3.2 Infinite-transition-rate Planning

For the purposes of this section, we assume that the switching process is irreducible; i.e. it is possible to eventually get to any mode starting from any other mode. Given the transition rate matrix  $\Lambda$ , it is logical to ask what portion of time the process spends in each mode. This gives rise to an invariant distribution on  $\mathcal{M}$ . We recall several of its standard properties (e.g., see **Theorem 3.5.3** and **Theorem 3.6.2** in [33]):

For an irreducible transition rate matrix  $\Lambda$  and its probability matrix  $P(t)$ ,

- (1) There exists a unique invariant distribution  $\pi^* \in \mathbb{R}^n$  satisfying  $\pi^* = \pi^* P(t)$  for all  $t > 0$ ;
- (2)  $\lim_{t \rightarrow +\infty} \|\pi P(t) - \pi^*\| = 0$  for all initial distributions  $\pi$ .

If transition rate matrix  $\Lambda$  is replaced by  $\Lambda_c = c\Lambda$  for some  $c > 0$ , the probability matrix becomes  $P_c(t) = \exp(ct\Lambda)$ . It is interesting to study the *infinite transition rate limit*: the behavior of the corresponding value functions  $u_i^c(\mathbf{x})$ 's as  $c \rightarrow \infty$ . Since every probability distribution  $\pi^c(t)$  will satisfy

$$\pi^c(t) = \pi^c(0)P_c(t) = \pi^c(0)\exp(ct\Lambda) = \pi^c(0)P_1(ct) = \pi^1(ct), \quad \lim_{c \rightarrow \infty} \pi^c(t) = \lim_{t \rightarrow \infty} \pi^1(t) = \pi^*,$$

in the limiting case the distribution is always  $\pi^*$  at any  $t > 0$  regardless of the initial mode. Additional technical assumptions (made precise in Theorem A.1 in Appendix), yield an upper bound for mode differences:

$$\|u_i^c(\mathbf{x}) - u_j^c(\mathbf{x})\|_\infty = O(c^{-1}). \quad (3.2)$$

Thus, in the infinite-transition-rate limit, the value functions in all modes will be the same, and we can simply use  $u^\infty(\mathbf{x})$  instead of  $u_i^\infty(\mathbf{x})$  to represent the value function. Additional controllability assumptions about  $\mathbf{f}_i$ 's will yield the local Lipschitz-continuity of all  $u_i^c$ 's on  $\Omega \setminus Q$ . An argument similar to the one in [22, 7] can be then used to prove that their convergence to  $u^\infty$  is uniform. The latter function can be recovered as the viscosity solution of the HJB PDE

$$\min_{\mathbf{a} \in A} \{ \nabla u^\infty(\mathbf{x}) \cdot \mathbb{E}_{i \sim \pi^*} [\mathbf{f}_i(\mathbf{x}, \mathbf{a})] + \mathbb{E}_{i \sim \pi^*} [C_i(\mathbf{x}, \mathbf{a})] \} = 0, \quad (3.3)$$

where  $\mathbb{E}_{i \sim \pi^*} [\mathbf{f}_i(\mathbf{x}, \mathbf{a})] = \sum_{i=1}^n \pi_i^* \mathbf{f}_i(\mathbf{x}, \mathbf{a})$  and  $\mathbb{E}_{i \sim \pi^*} [C_i(\mathbf{x}, \mathbf{a})] = \sum_{i=1}^n \pi_i^* C_i(\mathbf{x}, \mathbf{a})$ .

---

<sup>1</sup>An example of this kind is the description of one-wind-direction-only value functions  $v_i$  in the Introduction.

Due to the irreducibility of the switching process<sup>2</sup>, the choice of optimal controls in the limit becomes independent of the current mode  $i$  and can be determined based on  $\pi^*$ . We note that the optimal controls derived from (3.3) can be applied even for finite  $c$  as a way of planning for a *partially observable* process. (I.e., what is the best way to reach the target if we do not know the current mode  $m(t)$ ?) This approach is advocated in several prior papers [41, 13]. In section 5 we show that it has significant disadvantages for robotic path planning problems.

### 3.3 Using Miscalculated Transition Rates

Another interesting question is the expected total cost of path planning if we base it on “approximately correct” (rather than the real) transition rates. This arises naturally if the rates we use come from the statistics accumulated in previous runs. Alternatively, even if the real transition rate matrix  $\Lambda^r = (\lambda_{ij}^r)$  is known explicitly, one might prefer to replace it with  $\Lambda^p = (\lambda_{ij}^p)$  used for path planning purposes just for the sake of computational efficiency. Assuming that  $\Lambda^r$  and  $\Lambda^p$  are related by simple scaling as in subsection 3.2, we will use the corresponding  $c$  values (including the limiting case  $c = \infty$ ) in the superscript. If  $\alpha^p \in \mathcal{A}$  is the “optimal” control based on  $\Lambda^p$ , we define the expected cost  $u^{r,p}$  of using that control when  $\Lambda^r$  is in effect as

$$u_i^{r,p}(\mathbf{x}) = \mathbb{E}[J(\mathbf{x}, i, \alpha^p) | \Lambda^r]. \quad (3.4)$$

The case  $r = p$  corresponds to the equations in sections 2-3.2; otherwise,  $u^{r,p}$  can be recovered by solving a coupled system of linear HJB PDEs

$$\nabla u_i^{r,p}(\mathbf{x}) \cdot \mathbf{f}_i(\mathbf{x}, \alpha^p(\mathbf{x}, i)) - \sum_{j \neq i} \lambda_{ij}^r (u_i^{r,p}(\mathbf{x}) - u_j^{r,p}(\mathbf{x})) + C_i(\mathbf{x}, \alpha^p(\mathbf{x}, i)) = 0, \quad (3.5)$$

where  $\alpha^p(\mathbf{x}, i) = \arg \min_{\mathbf{a} \in \mathcal{A}} \{\nabla u_i^{r,p}(\mathbf{x}) \cdot \mathbf{f}_i(\mathbf{x}, \mathbf{a}) + C_i(\mathbf{x}, \mathbf{a})\}$  are assumed to be already known.

## 4 Numerical Method

To simplify the notation, we will focus on the time-optimal control problems (i.e.,  $C_i(\mathbf{x}, \mathbf{a}) = 1$  for all  $\mathbf{x}, \mathbf{a}$ , and  $i$ ). The numerical solution is sought on a Cartesian grid with grid spacing  $h$  imposed over  $\Omega$ . We will use  $\mathcal{X}$  to denote the set of all gridpoints. A separate copy of the grid is used for each mode  $i \in \mathcal{M}$ . The PDE in a continuous domain is replaced by a coupled system of discretized equations (one for each  $\mathbf{z} = (\mathbf{x}, i)$ ). So, the approximation method consists of two components:

1. a formula for approximating each individual  $u(\mathbf{x}, i)$  if the value function for each neighboring gridpoint in  $\mathcal{X} \times \mathcal{M}$  is already known;
2. an algorithm for solving the entire coupled system.

We start with the latter, which is largely “discretization-neutral”, and postpone the former until subsection 4.2. The following notations will be used throughout the section:

---

<sup>2</sup> For a more general case of possibly reducible switching processes, the value function does depend on the *communicating class* of the starting mode [33], and a PDE similar to (3.3) would have to be solved for each communicating class.

$h$	the gridline spacing;
$\mathcal{X}$	the set of all gridpoints in $\Omega$
$\mathcal{Z}$	the set of all gridpoints in $\Omega \times \mathcal{M}$
$X(\mathbf{z})$	the projection onto $\mathcal{X}$ for all $\mathbf{z} \in \mathcal{Z}$
$M(\mathbf{z})$	the projection onto $\mathcal{M}$ for all $\mathbf{z} \in \mathcal{Z}$
$U(\mathbf{z})$	the approximate value function at $\mathbf{z}$
$\mathcal{I}(\mathbf{z})$	the (discretization-dependent) set of “possibly influenced” gridpoints; i.e., $\mathcal{I}(\mathbf{z}) = \{\hat{\mathbf{z}} \in \mathcal{Z} \mid U(\hat{\mathbf{z}}) \text{ might depend on } U(\mathbf{z})\}$
$\mathcal{D}(\mathbf{z})$	the (discretization-dependent) set of “possibly influencing” gridpoints; i.e., $\mathcal{D}(\mathbf{z}) = \{\hat{\mathbf{z}} \in \mathcal{Z} \mid \mathbf{z} \in \mathcal{I}(\hat{\mathbf{z}})\}$
$\mathcal{Q}$	the discretized target set; i.e., $\mathcal{Q} = \mathcal{X} \cap Q$
$q(\mathbf{x})$	the boundary condition for $\mathbf{x} \in \mathcal{X}$
$\mathcal{N}(\mathbf{x})$	the set of neighboring gridpoints; i.e., $\mathcal{N}(\mathbf{x}) = \{\bar{\mathbf{x}} \in \mathcal{X} \mid \ \mathbf{x} - \bar{\mathbf{x}}\  = h\}$
$ActiveFlag(\mathbf{z})$	a boolean flag indicating whether $u(\mathbf{z})$ should be recomputed

Whenever the correspondence  $\mathbf{z} = (\mathbf{x}, i)$  is clear from the context, we will use both the  $\mathbf{z}$ -based and  $(\mathbf{x}, i)$ -based notations interchangeably; e.g.,  $U(\mathbf{x}, i) = U(\mathbf{z})$ , and  $\mathbf{f}_i(\mathbf{x}, \mathbf{a}) = \mathbf{f}(\mathbf{z}, \mathbf{a})$ .

We will focus on discussing algorithms for the general weakly-coupled planning. The problems described in sections 3.1 and 3.2 can be handled similarly, but are also covered by more efficient non-iterative methods; e.g., [39, 2, 32].

## 4.1 Value Iterations

Our overall approach is iterative – a Gauss-Seidel relaxation of the *value iterations method*. This is essentially an extension of *fast sweeping methods*, originally introduced for single-mode problems [11, 42]. We alternate through a set of predefined “geometric orderings” to sweep through gridpoints in  $\mathcal{X}$  (e.g., in 2D, the four sweeping directions are from the south-west, from the south-east, from the north-east, and from the north-west). In each sweep, we attempt to update the value at each gridpoint by solving the discretized HJB equations (see section 4.2); whenever a specific  $\mathbf{x} \in \mathcal{X}$  is processed, we re-compute  $U(\mathbf{x}, i)$  for all  $i \in \mathcal{M}$ . The newly-computed value is only used if it is smaller than the previous version of  $U(\mathbf{x}, i)$ . This relies on the *monotonicity* of our discretization (discussed in section 4.2) and the initialization  $U(\mathbf{x}, i) = +\infty$  for all  $\mathbf{x} \notin \mathcal{Q}$ .

Our implementation also relies on one further speed-up technique: *the active flags*, first used in [4] for the Eikonal PDE, are employed to identify the gridpoints that might need to be updated in each sweep. (There is no point in updating  $\mathbf{z}$  if none of the values on  $\mathcal{D}(\mathbf{z})$  have changed since the last time  $U(\mathbf{z})$  was computed.) Finally, the iterations are terminated once the maximal change in the value function observed in the last sweep falls below the predefined tolerance parameter  $\varepsilon > 0$ .

**Remark 4.1.** In the current loop structure (iterate over  $\mathcal{M}$  in the inner loop), none of the marching-type methods (e.g., [37, 39]) are directly applicable since characteristic directions are usually quite different in different modes and the value functions for different modes are coupled. One could also use an alternative loop structure, iterating over  $\mathcal{M}$  in the outer loop. This technique iteratively “freezes” the value function for all modes but one, with the latter then solved by the same techniques available for decoupled problems. All previously developed methods (marching, sweeping, hybrid, label-setting, etc) are available as solvers for this “one mode updated at a time” problem. However, our computational experiments showed that this alternative loop structure usually requires a much larger number of iterations. This determined our implementation choices described in Algorithm 4.1.

---

**Algorithm 1:** Pseudocode for solving the dynamic programming equations on a grid.

---

```

1 #Initialization:
2 for  $\mathbf{z} \in \mathcal{Z}$  do
3   |  $ActiveFlag(\mathbf{z}) \leftarrow \text{false}$ 
4 end
5 for  $\mathbf{x} \in \mathcal{X}$  do
6   | if  $\mathbf{x} \in \mathcal{Q}$  then
7     |   for  $i \in \mathcal{M}$  do
8       |      $U(\mathbf{x}, i) \leftarrow q(\mathbf{x})$ 
9       |     for  $\hat{\mathbf{z}} \in \mathcal{I}(\mathbf{x}, i)$  do
10        |       |  $ActiveFlag(\hat{\mathbf{z}}) \leftarrow \text{true}$ 
11        |     end
12      end
13    else
14      |   for  $i \in \mathcal{M}$  do
15        |      $U(\mathbf{x}, i) \leftarrow \infty$ 
16      end
17    end
18 end
19
20 #Main Loop:
21  $converged \leftarrow \text{false}$ 
22 while not converged do
23    $max\_change \leftarrow 0$ 
24   for  $\mathbf{x} \in \mathcal{X} \setminus \mathcal{Q}$  enumerated in the current sweep order do
25     | for  $i \in \mathcal{M}$  do
26       |    $\mathbf{z} \leftarrow (\mathbf{x}, i)$ 
27       |   if ( $ActiveFlag(\mathbf{z}) == \text{true}$ ) then
28         |      $ActiveFlag(\mathbf{z}) \leftarrow \text{false}$ 
29         |      $U^* \leftarrow \text{update}(\mathbf{z})$ 
30         |      $change\_z \leftarrow U(\mathbf{z}) - U^*$ 
31         |      $max\_change \leftarrow \max\{change\_z, max\_change\}$ 
32         |      $U(\mathbf{z}) \leftarrow U^*$ 
33         |     for  $\hat{\mathbf{z}} \in \mathcal{I}(\mathbf{z})$  do
34           |       | if  $X(\hat{\mathbf{z}}) \notin \mathcal{Q}$  then
35             |         |  $ActiveFlag(\hat{\mathbf{z}}) \leftarrow \text{true}$ 
36             |       end
37           end
38         end
39       end
40       if  $max\_change < \varepsilon$  then
41         |  $converged \leftarrow \text{true}$ 
42       else
43         | switch to the next sweep order
44       end
45     end
46 end

```

---

# subsection 4.2



## 4.2 Discretizations on the Grid

We discuss several first-order accurate update formulas used to compute  $U(\mathbf{z})$  if  $U(\hat{\mathbf{z}})$  are already known for all  $\hat{\mathbf{z}} \in \mathcal{D}(\mathbf{z})$ . We start with the most general semi-Lagrangian discretization and then show how additional assumptions about the dynamics can be leveraged to obtain more efficient algorithms. We introduce the notation used throughout this section:

$(\mathbf{e}_1, \dots, \mathbf{e}_d)$	the canonical basis in $\mathbb{R}^d$
$\text{ort} = (\epsilon_1, \dots, \epsilon_d)$	a vector encoding a particular orthant in $\mathbb{R}^d$ : $\epsilon_k = \pm 1, \quad k = 1, \dots, d$
$\mathcal{O}$	the set of all orthants in $\mathbb{R}^d$
$\xi = (\xi_1, \dots, \xi_d)$	a point in the unit $(d-1)$ -simplex: $\sum_{k=1}^d \xi_k = 1, \xi_k \geq 0, \quad k = 1, \dots, d$
$\Xi$	the unit $(d-1)$ -simplex

### 4.2.1 Semi-Lagrangian Method

Semi-Lagrangian discretizations are based on following the characteristic for a short time and then using the interpolated value function at the resulting point. A comprehensive overview for the fully deterministic case can be found in [24].

The idea is to use the dynamic programming equation (2.7) to approximate the value function:

$$U(\mathbf{x}, i) \approx \min_{\mathbf{a} \in A} \left\{ \tau + \sum_{j \in \mathcal{M}} p_{ij}(\tau) U(\tilde{\mathbf{x}}_{\mathbf{a}, \tau}, j) \right\}, \quad \tilde{\mathbf{x}}_{\mathbf{a}, \tau} = \mathbf{x} + \tau \mathbf{f}_i(\mathbf{x}, \mathbf{a}). \quad (4.1)$$

$U(\tilde{\mathbf{x}}_{\mathbf{a}, \tau}, j)$  is computed by an interpolation procedure, since  $\tilde{\mathbf{x}}_{\mathbf{a}, \tau}$  is generally not a gridpoint. Here, we can use a first order approximation for  $p_{ij}(\tau)$ .

The discretization also implicitly depends on the choice of the (pseudo-)timestep  $\tau$ . One popular choice is to take  $\tau > 0$  to be a constant proportional to  $h$  [23]. In this approach, it is common to approximate  $U(\tilde{\mathbf{x}}_{\mathbf{a}, \tau}, j)$  through bi-linear interpolation using all  $2^d$  vertices of the grid cell containing  $(\tilde{\mathbf{x}}_{\mathbf{a}, \tau}, j)$ .

Another choice is to use a control-dependent  $\tau_{\mathbf{a}} = h / \|\mathbf{f}_i(\mathbf{x}, \mathbf{a})\|$  to ensure that  $\tilde{\mathbf{x}}_{\mathbf{a}, \tau}$  lies in a  $(d-1)$ -dimensional simplex formed by the gridpoints adjacent to  $\mathbf{x}$ . In this case, we can interpolate  $U(\tilde{\mathbf{x}}_{\mathbf{a}, \tau}, j)$  linearly using the values at the vertices of that simplex.

In our implementation we use the latter approach, which has important computational advantages (discussed at the end of this section), particularly for the *small-time controllable problems*, where the controller can move the system in every direction regardless of the current mode. I.e., for the rest of this section we will assume that, given any  $\forall \mathbf{z} \in \mathcal{Z}$  and any  $(\text{ort}, \xi) = ((\epsilon_1, \dots, \epsilon_d), (\xi_1, \dots, \xi_d)) \in \mathcal{O} \times \Xi$ , there exists a unique control  $\mathbf{a} = \mathbf{a}(\mathbf{z}, \text{ort}, \xi) \in A$  and  $\tau = \tau(\mathbf{z}, \text{ort}, \xi) > 0$  such that

$$\tau \mathbf{f}(\mathbf{z}, \mathbf{a}) = (h\epsilon_1\xi_1, \dots, h\epsilon_d\xi_d). \quad (4.2)$$

Starting from  $\mathbf{z} \in \mathcal{Z}$  with the control  $\mathbf{a} = \mathbf{a}(\mathbf{z}, \text{ort}, \xi)$  determined by  $(\text{ort}, \xi)$ , we can compute the corresponding timestep

$$\tau = \tau(\mathbf{z}, \text{ort}, \xi) = \frac{\sqrt{\sum_{k=1}^d \xi_k^2}}{\|\mathbf{f}(\mathbf{z}, \mathbf{a})\|} h, \quad (4.3)$$

and define the set of contributing neighbors

$$\mathbf{x}_k = \mathbf{x} + \epsilon_k h \mathbf{e}_k, \quad k = 1, \dots, d.$$

If  $(\text{ort}, \xi)$  encodes the optimal direction, the approximate value function at  $\mathbf{z}$  is

$$\tilde{U}(\mathbf{z}, \text{ort}, \xi) = \tau + \sum_{k=1}^d \xi_k \left( p_{ii}(\tau) U(\mathbf{x}_k, i) + \sum_{j \neq i} p_{ij}(\tau) U(\mathbf{x}_k, j) \right). \quad (4.4)$$

Therefore, instead of searching for the optimal control  $\mathbf{a}^* \in A$ , we can search for the optimal  $(\text{ort}, \xi)^* \in \mathcal{O} \times \Xi$ . The minimization is efficiently performed on an orthant-by-orthant basis; i.e.,

$$U(\mathbf{z}) = \min_{(\text{ort}, \xi) \in \mathcal{O} \times \Xi} \tilde{U}(\mathbf{z}, \text{ort}, \xi). \quad (4.5)$$

This update strategy is summarized in Algorithm 2. Since  $\xi_k$  and  $p_{ij}(\tau)$  are always non-negative, (4.4-4.5) ensure that  $U(\mathbf{z})$  is a monotone non-decreasing function of  $U(\mathbf{x}_k, j)$  for all  $k$  and  $j$ . As a result, the updates in Algorithm 4.1 are also performed in a monotone fashion; i.e., all gridpoint values are decreasing.

---

**Algorithm 2:** A semi-Lagrangian update function with a control-dependent timestep.

---

```

1 function  $U^* = \text{update}(\mathbf{z})$ :
2    $U^* \leftarrow U(\mathbf{z})$ 
3   for  $\text{ort} = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_d) \in \mathcal{O}$  do
4      $U' \leftarrow \min_{\xi \in \Xi} \tilde{U}(\mathbf{z}, \text{ort}, \xi)$ 
5     (see formula (4.4))
6
7     if  $U' < U^*$  then
8        $U^* \leftarrow U'$ 
9     end
10  end
11 return  $U^*$ 

```

---

#### 4.2.2 Eulerian Discretization

In general, the minimization in (4.4) has to be performed numerically, but for special types of dynamics it might be possible to find an analytic formula for the optimal direction, and Eulerian numerical schemes become preferable. To illustrate the latter, we will focus on the dynamics of the rowboat affected by changing winds (as described in section 1). The boat velocity in mode  $i \in \mathcal{M}$  is  $\mathbf{f}_i(\mathbf{x}, \mathbf{a}) = s(\mathbf{x})\mathbf{a} + \mathbf{w}_i(\mathbf{x})$ , where  $\mathbf{a} \in \mathbb{S}^1$  is our rowing direction,  $s(\mathbf{x})$  is the rowing speed, and  $\mathbf{w}_i$  is the velocity component due to the  $i$ -th wind map. In this case,

$$\min_{\mathbf{a} \in A} \{\nabla u(\mathbf{x}, i) \cdot \mathbf{f}_i(\mathbf{x}, \mathbf{a})\} = -s(\mathbf{x}) \|\nabla u(\mathbf{x}, i)\| + \nabla u(\mathbf{x}, i) \cdot \mathbf{w}_i(\mathbf{x}). \quad (4.6)$$

Therefore, the HJB equation for  $u(\mathbf{x}, i)$  becomes

$$s(\mathbf{x}) \|\nabla u(\mathbf{x}, i)\| = \nabla u(\mathbf{x}, i) \cdot \mathbf{w}_i(\mathbf{x}) - \sum_{j \neq i} \lambda_{ij} (u(\mathbf{x}, i) - u(\mathbf{x}, j)) + 1. \quad (4.7)$$

To simplify the notation, we will only describe the discretization for  $d = 2$ . Having chosen a particular quadrant  $\text{ort} = (\epsilon_1, \epsilon_2)$ , we can use

$$D^h U(\mathbf{x}, i) = \left[ \frac{U(\mathbf{x} + \epsilon_1 h \mathbf{e}_1, i) - U(\mathbf{x}, i)}{\epsilon_1 h}, \frac{U(\mathbf{x} + \epsilon_2 h \mathbf{e}_2, i) - U(\mathbf{x}, i)}{\epsilon_2 h} \right]^T$$

to approximate  $\nabla u(\mathbf{x}, i)$ . Plugging in this approximation into (4.7), reduces the PDE to

$$s^2(\mathbf{x}) \|D^h U(\mathbf{x}, i)\|^2 = \left[ D^h U(\mathbf{x}, i) \cdot \mathbf{w}_i(\mathbf{x}) - \sum_{j \neq i} \lambda_{ij} (U(\mathbf{x}, i) - U(\mathbf{x}, j)) + 1 \right]^2. \quad (4.8)$$

Assuming the values of  $U$  at all neighboring gridpoints in  $\mathbf{Z}$  are known, this is simply a quadratic equation for  $U(\mathbf{x}, i)$ .

Note that  $\mathbf{a}^* = \frac{-D^h U}{\|D^h U\|}$  approximates the optimal control value at  $(\mathbf{x}, i)$ . If the quadratic equation (4.8) has real roots, we need to satisfy the additional *upwinding condition*, by choosing the smallest root for which the velocity vector points from the same quadrant used to approximate  $\nabla u(x, i)$ ; i.e.,

$$\mathbf{f}_i(\mathbf{x}, \mathbf{a}^*) \cdot \epsilon_k \mathbf{e}_k \geq 0, \quad k = 1, 2. \quad (4.9)$$

A straightforward computation shows that, if the quadratic equation (4.8) has more than one real root, the smaller root is always inconsistent with the upwinding condition (4.9), and only the larger of them is relevant.

If the quadratic equation does not have real roots, or the upwinding condition (4.9) is not satisfied, we default to “one-sided” updates along the directions  $\epsilon_k \mathbf{e}_k$  for  $k = 1, 2$ :

$$U(\mathbf{x}, i) = \min_{k=1,2} \tilde{U}_k, \quad \text{with} \quad \tilde{U}_k = \frac{\tau_k + U(\mathbf{x} + h\epsilon_k \mathbf{e}_k, i) + \tau_k \sum_{j \neq i} \lambda_{ij} U(\mathbf{x}, j)}{1 + \tau_k \sum_{j \neq i} \lambda_{ij}}, \quad (4.10)$$

where  $\tau_k = \frac{h}{\mathbf{f}_i(\mathbf{x}, \mathbf{a})}$  and  $\mathbf{a} \in A$  is such that  $\mathbf{f}_i(\mathbf{x}, \mathbf{a}) = c\epsilon_k \mathbf{e}_k$  for some  $c > 0$ . This approximation is derived by taking functions  $u(\mathbf{x}, j)$  to be known for all  $j \neq i$  and writing down the characteristic ODEs for  $u(\mathbf{x}, i)$  with the assumption that the characteristic direction at  $\mathbf{x}$  is  $(\epsilon_k \mathbf{e}_k)$ .

Finally, since the entire discretization was described for one quadrant, we need to minimize the update over all  $\text{ort} \in \mathcal{O}$ . The complete update procedure is summarized in Algorithm 3.

---

**Algorithm 3:** An Eulerian update function.

---

```

1 function  $U^* = \text{update}(\mathbf{z})$ :
2  $U^* \leftarrow U(\mathbf{z})$ 
3 for  $\text{ort} = (\epsilon_1, \epsilon_2) \in \mathcal{O}$  do
4   if (4.8) has real roots AND the larger root  $\tilde{U}$  satisfies the upwinding condition (4.9) then
5      $U^* \leftarrow \min(U^*, \tilde{U})$ 
6   else
7     for  $k = 1, 2$  do
8       compute  $\tilde{U}_k$  by formula (4.10)
9        $U^* \leftarrow \min(U^*, \tilde{U}_k)$ 
10    end
11  end
12 end
13 return  $U^*$ 

```

---

Even though the approach described here is fairly similar to what is done in the Eikonal/single-mode case, there are several important subtleties worth noting.

1. This discretization is first order accurate in  $h$ , consistent, and monotone. I.e., it is easy to verify that  $U(\mathbf{x}, i)$  is a monotone non-decreasing function of all  $U(\mathbf{x}, j)$  and  $U(\mathbf{x} + h\epsilon_k \mathbf{e}_k, i)$ . When  $u(\mathbf{x}, j)$  is known for all  $j \neq i$ , the standard argument [6] shows the convergence of  $U(\mathbf{x}, i)$  to the viscosity solution  $u(\mathbf{x}, i)$ . This is also the reason why the update is only used if it is below the previously computed value of  $U(\mathbf{x}, i)$ . (Recall that all  $U$  values outside of  $\mathcal{Q}$  are initialized to  $\infty$  in Algorithm 4.1.)
2. The upwinding condition (4.9) is quite different from the one imposed in the simple Eikonal case. In particular,  $U(\mathbf{x}, i)$  generally *is not* larger than the neighboring values from the quadrant used to compute it. This means that the Fast Marching Method [37] is not directly applicable. An attempt to extend it to this case has been presented in [19] using a two-sided update equivalent to (4.8); however, the lack of causality prevents the convergence for a wide range of examples. The causal properties can be restored by sufficiently extending the stencil (as in Ordered Upwind Methods) and the corresponding examples were presented in [39]. Similar problems were also handled by Fast Sweeping in [18].
3. An extension of (4.8) and (4.9) to higher dimensional problems is quite straightforward. However, when the upwinding condition is not satisfied, it becomes necessary to solve a sequence of quadratic equations corresponding to characteristics lying in lower dimensional faces of the update orthant.

### 4.2.3 Comparison of discretizations

We finish this section by summarizing the properties of three discretizations discussed above: (a) the semi-Lagrangian method with a constant timestep  $\tau$ , (b) the semi-Lagrangian method with a control-dependent  $\tau$ , and (c) the Eulerian method. All of them are first-order accurate, but their computational costs are quite different.

- Method (a) works on the most general problems, but is more computationally expensive than (b) and (c) because (1) the number of iterations required for convergence is large when  $\tau$  is relatively small; (2) a larger stencil makes it more difficult to impose the boundary conditions and treat exit sets with empty interior; (3) the non-local influence/dependence sets (i.e.,  $\mathcal{I}(\mathbf{z})$  and  $\mathcal{D}(\mathbf{z})$ ) make it far more difficult to implement ActiveFlags efficiently.
- Method (b) is more computationally efficient and often more accurate due to its stencil locality ( $\mathcal{I}(\mathbf{x}, i) = \mathcal{D}(\mathbf{x}, i) = \mathcal{N}(\mathbf{x}) \times \mathcal{M}$ ). However, it still requires numerical minimization and is only suitable for speed profiles containing the origin in the interior.
- Method (c) is the most efficient of these three, but it is only applicable when the optimal direction can be found explicitly as a function of the gradient. Its computational stencil is also local but slightly different; i.e.,  $\mathcal{I}(\mathbf{x}, i) = \mathcal{D}(\mathbf{x}, i) = (\mathcal{N}(\mathbf{x}) \times \{i\}) \cup (\{\mathbf{x}\} \times \mathcal{M})$ . An argument based on Kuhn-Tucker optimality conditions can be used to show its equivalence to a related semi-Lagrangian scheme, and its output is at most  $O(\tau^2)$  different from the numerical solution produced by the method (b). We omit the proof for the sake of brevity; see the Appendix in [3] for a similar approach.

## 5 Experimental Results

Our experiments are based on a simple model problem described in section 1: a rowboat trying to reach its destination in the presence of (randomly changing) wind. We also compare the performance of optimal controls with those found under simplifying assumptions (i.e., zero or infinite wind-switching rate).

## 5.1 Experimental setting and parameter values

We minimize the expected time for the boat whose velocity is defined by

$$\mathbf{f}_i(\mathbf{x}, \mathbf{a}) = s(\mathbf{x})\mathbf{a} + \mathbf{w}_i(\mathbf{x}), \quad \mathbf{a} \in \mathbb{S}^1, \quad i = 1, \dots, n,$$

with the assumption that  $\|\mathbf{w}_i\| < s(\mathbf{x})$ , for all  $(\mathbf{x}, i)$ . If the transition between the modes are possible, this results in a weak-coupling and yields the system of equations (4.7), which we solve iteratively using the Eulerian discretization from section 4.2.2 on a cartesian grid with  $h = \frac{1}{320}$ . The iterations are terminated when the changes in grid values become lower than  $\varepsilon = 10^{-6}$ .

The emphasis of our experiments is on the influence of mode-switching rates on the optimal controls. To this end, we further simplify the dynamics by taking the constant boat-speed in still water ( $s(\mathbf{x}) = 2$ ), and two constant wind map versions:

$$n = 2, \quad \mathbf{w}_1(\mathbf{x}) = [1.5, 0]^T, \quad \mathbf{w}_2(\mathbf{x}) = [-1.5, 0]^T.$$

We restrict the dynamics to a square domain with a rectangular obstacle

$$\Omega = (0, 1) \times (0, 1) \setminus [0.1, 0.85] \times [0.1, 0.15]$$

and choose the target at the location  $Q = \{(0.5, 0.05)\}$ . The boundary conditions are specified by  $q = 0$  on  $Q$  and  $q = +\infty$  on  $\partial\Omega \setminus Q$ .

Finally, we also make the mode switching symmetric by taking  $\lambda_{12} = \lambda_{21} = \lambda$  and study the change in the value function as we vary this  $\lambda$ .

## 5.2 Value Functions

Figure 1 shows the value functions computed for  $\lambda = 0, 1, 10, 50$  and  $\infty$  with a magenta disk used to indicate the target. When  $\lambda = 0$ , the problem corresponds to the uncoupled planning problem in (3.1). When  $\lambda = \infty$ , the problem becomes the infinite-transition-rate planning problem in (3.3). The computational cost of our experiments is very dependent on the strength of coupling: the number of iterations/sweeps needed up to convergence is 6, 19, 35, 87 and 6 for the respective  $\lambda$  values listed above.

In each subfigure there is a clearly visible “shockline” above the obstacle, where the gradient of the value function is undefined. The optimal trajectory runs clockwise or counterclockwise around the obstacle depending on whether our starting position is to the right or to the left of the shockline. The optimal direction is not unique on the shockline with both approaches yielding the same (expected) time to the target for these starting locations. It is important to note that the locations of shocklines are mode and  $\lambda$ -dependent. When  $\lambda = 0$ , the shocklines of both modes are very sharp and almost in the center of the figure. As  $\lambda$  increases, the shocklines move left or right (depending on the mode) away from the center and become less pronounced. (E.g., for  $\lambda = 10$  and most starting positions sufficiently far North of the obstacle, it appears to be optimal to row South, directly toward the obstacle, instead of aiming for one of its corners. The decision on whether to go clockwise or counterclockwise is postponed until we are closer to the obstacle.) But when  $\lambda$  becomes even higher, the shocklines are sharp again and return to the center.

We also use these computations to verify that the value functions of different modes become more similar as  $\lambda$  increases. Figure 2 is fully in agreement with the asymptotic rate (3.2) and the upper bound (A.3).

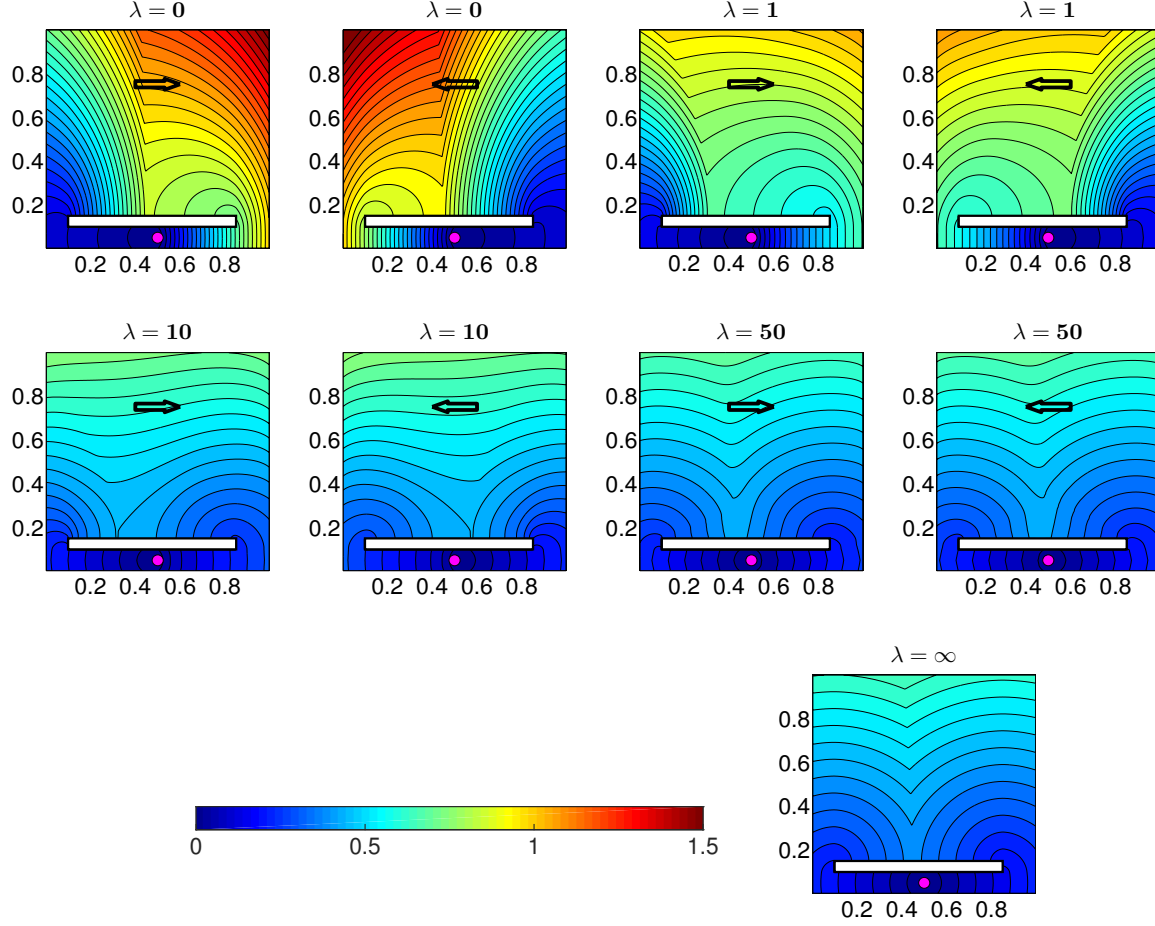


Figure 1: The value functions for  $\lambda = 0$ ,  $\lambda = 1$ ,  $\lambda = 10$ ,  $\lambda = 50$  and  $\lambda = \infty$ . Wind directions represented by arrows.

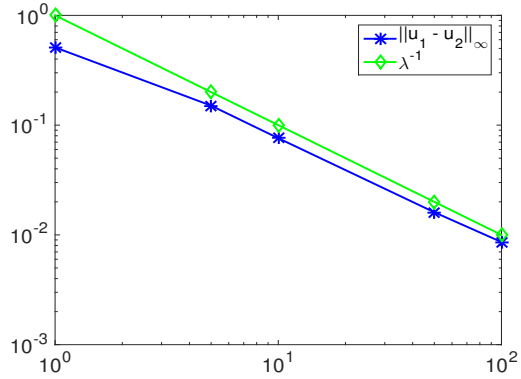


Figure 2: The mode differences  $\|u_1 - u_2\|_\infty$  plotted for  $\lambda = 1, 5, 10, 50, 100$ . The case  $\lambda = 0$  is not plotted, but  $\|u_1^0 - u_2^0\|_\infty \approx 0.8518$ .

### 5.3 Path Planning and Simulations

We perform a series of Monte-Carlo trials for trajectories starting from the same position/mode, but using randomly generated sequences of mode-switching times.

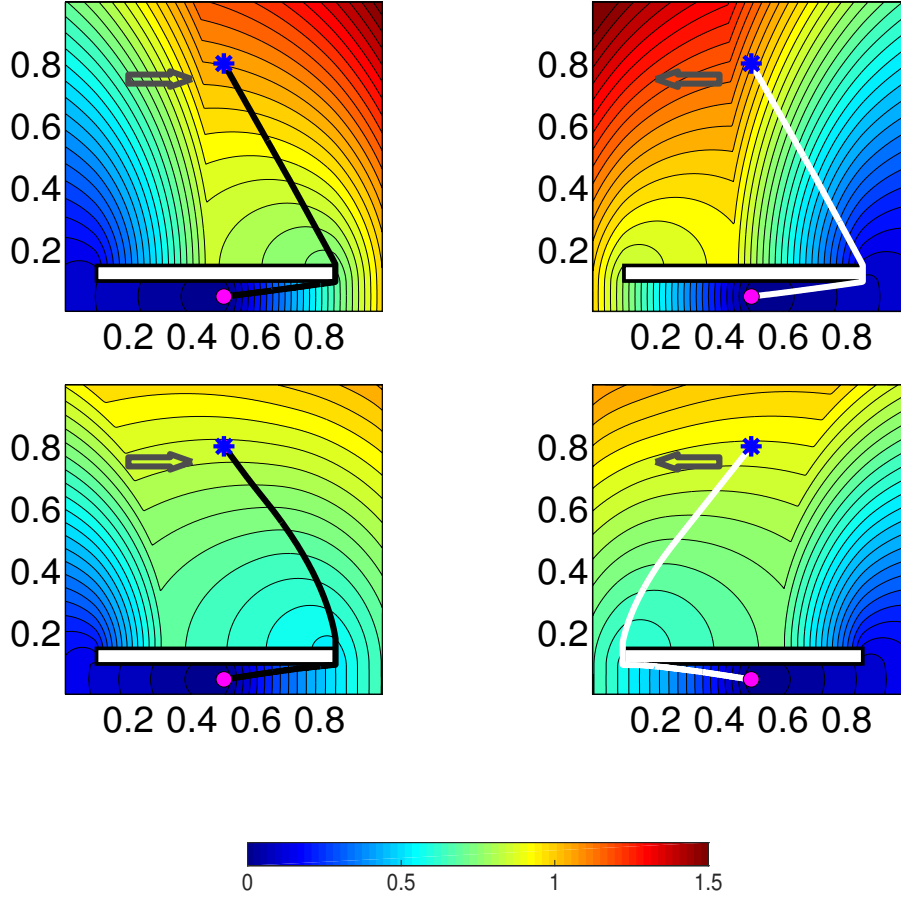


Figure 3: “Coupled” vs “uncoupled” path planning. A reference trajectory shown for the case when no wind transitions happen before we reach the target. The color coding of trajectories corresponds to the current wind direction: black is to the East while white is to the West. The first row: uncoupled planning (based on  $u_i^0(\mathbf{x})$ ). The second row: coupled planning under the assumption that  $\lambda_{12} = \lambda_{21} = 1$ .

Once the rate of switching  $\lambda > 0$  is fixed, we can compute the value function and define the optimal feedback policy  $\alpha_*(\mathbf{x}, i)$  by using the minimizing value  $\mathbf{a}$  in equation (2.9). However, our current goal is to explore the performance changes due to the use of asymptotic approximations: both  $u^\infty(\mathbf{x})$  and  $u_i^0(\mathbf{x})$  are cheaper to compute than  $u_i(\mathbf{x})$  and define corresponding “optimal” feedback policies ( $\alpha_*^\infty$  and  $\alpha_*^0$  respectively). In the path planning simulations, these policies can be used despite the fact that the random mode-switching times correspond to our specific  $\lambda$ . Since we are discussing the discrepancy of “assumed” and “real”  $\lambda$  values, it is natural to adopt the notation of section 3.3: we will refer to  $u$  as  $u^r$ , while  $u^0$  and  $u^\infty$  correspond to  $u^p$ . Equation (3.5) can be used to recover the expected time-to-target  $u^{r,p}$  resulting from this approximate planning. However, we prefer to conduct Monte-Carlo simulations instead, since they provide more information about the result: approximating the actual time-of-arrival distribution rather than just the expectation.

In our experiments, the starting point is always at  $\hat{\mathbf{x}} = (0.5, 0.8)$ , and the trajectories are color coded based on the current wind direction: black is for eastward while white is for westward. Given the current position and mode,  $(\mathbf{x}, i) \in \Omega \times \mathcal{M}$ , we use a constant control  $\alpha_*(\mathbf{x}, i)$  over the next  $\Delta t$  seconds. (We have used  $\Delta t = 10^{-3}$ , which is sufficiently small for the  $\lambda$  values 1 and 10 considered in

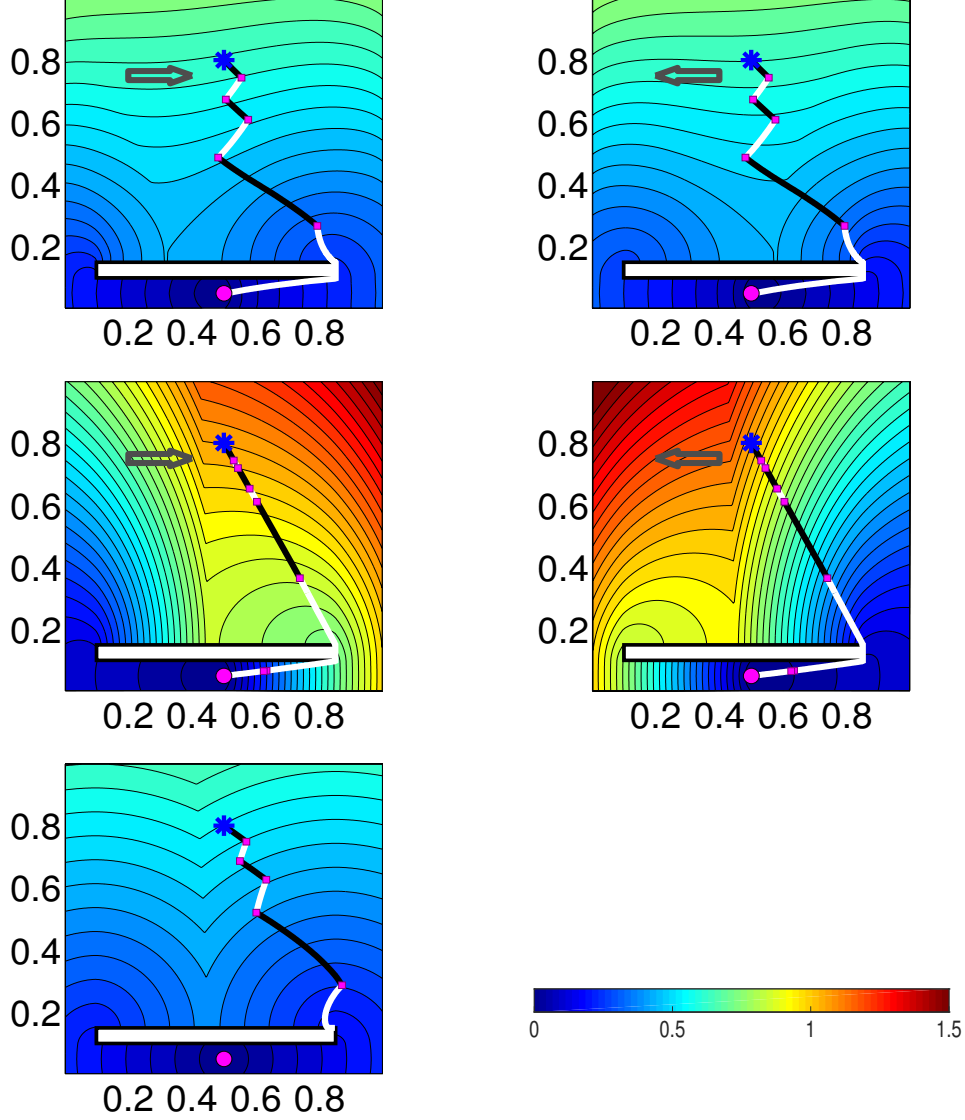


Figure 4: Trajectories based on coupled, uncoupled and infinite transition rate planners. The blue asterisk is the starting position  $\hat{\mathbf{x}}$ , and the initial wind direction is to the East. The magenta disk is the target. Wind directions represented by grey arrows. The same sequences of wind-switching times (based on the “real” transition rate  $\lambda_{12} = \lambda_{21} = 10$ ) used in all cases. The wind switch locations are shown by purple squares. The color coding of trajectories corresponds to the current wind direction: black is to the East while white is to the West. The first row: trajectory planning based on the correct/real transition rates. The second row: “uncoupled” planning (based on a no-switches assumption). The third row: infinite-transition-rate planning (we no longer keep track of the current mode) – the target is never reached since we collide with an obstacle.



these tests.) The position is shifted by  $\Delta t \mathbf{f}_i(\mathbf{x}, \boldsymbol{\alpha}_*(\mathbf{x}, i))$  and the mode possibly changes based on  $\lambda$ . The process is repeated until we reach the target. Figure 3 shows the trajectories without any mode switches based on the  $\boldsymbol{\alpha}_*^0$  planning and  $\boldsymbol{\alpha}_*$  planning with  $\lambda = 1$ . Since the Hamiltonians are homogeneous in  $\mathbf{x}$ , the trajectories resulting from  $\boldsymbol{\alpha}_*^0$  are piecewise-straight lines. For this particular starting location, it is optimal to go clockwise around the obstacle even if the wind blows eastward – since the counterclockwise alternative path is slightly longer (the obstacle is off center). However, if we take the correct switching rate into account, use  $\boldsymbol{\alpha}_*$  planning and start with the wind blowing eastward, it makes sense to go counterclockwise. The intuitive reason is that the wind is likely to switch toward West before we reach the target making our motion on the final stretch much faster. Interestingly, the trajectory we follow in anticipation of this switch is not piecewise straight. This is entirely due to a weak coupling of the PDEs since the Hamiltonians remain the same. However, this logic backfires in the particular scenario illustrated by Figure 3: since the wind never changes, the actual time along the trajectory based on  $\boldsymbol{\alpha}_*$  with the wind blowing eastward is  $T^\lambda = 1.179$ , which is larger than our expected  $E[T^\lambda] = u_2^\lambda(\hat{\mathbf{x}}) = 0.915$  and certainly larger than the no-switch optimal  $u_2^0(\hat{\mathbf{x}}) = 1.073$ . Luckily, this outcome is not the most likely: the probability of the wind direction remaining the same that long is  $\exp(-1.179) \approx 31\%$ .

Of course, there can be many mode switches happening before we reach the target, particularly when  $\lambda$  is larger. Figure 4 illustrates the results of a single simulation with many mode switchings based on  $\boldsymbol{\alpha}_*$ ,  $\boldsymbol{\alpha}_*^0$ , and  $\boldsymbol{\alpha}_*^\infty$  path planning. Using  $\lambda = 10$ , we have generated a sequence of switching times: (0.029, 0.064, 0.098, 0.159, 0.285, 0.689, 0.706, ...). With the correct feedback optimal controls  $\boldsymbol{\alpha}_*$ , the target is reached at the time  $T^\lambda = 0.589$ . With the uncoupled “optimal” control  $\boldsymbol{\alpha}_*^0$ , two more mode switches occur by the new arrival time  $T^0 = 0.741$ . Finally, if we use  $\boldsymbol{\alpha}_*^\infty$ , the target is not reached at all since a collision with an obstacle happens at the time  $t = 0.423$ . This outcome is not uncommon when using the infinite-transition-rate planning despite the fact that the real value of  $\lambda$  is finite. In our experiments based on  $\boldsymbol{\alpha}_*^\infty$ , collisions occurred in 22.5% of simulations for  $\lambda = 1$  and in 42.5% of simulations for  $\lambda = 10$ .

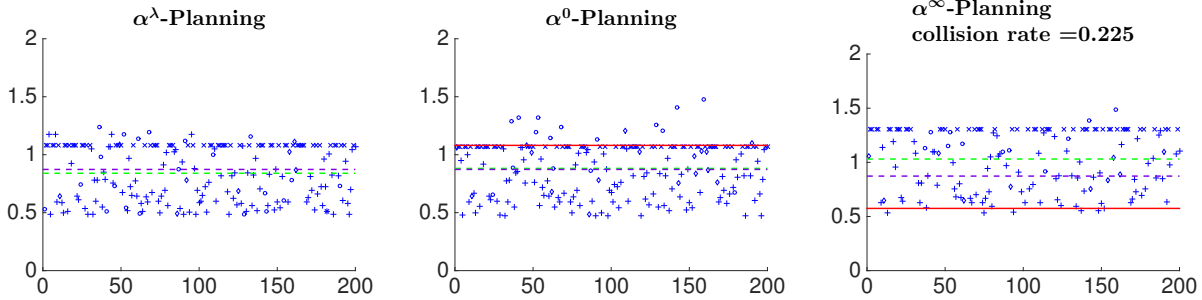


Figure 5: 200 simulations starting at  $((0.5, 0.8), 1)$  when real transition rate  $\lambda = 1$ .  $x$  axis: the number of simulations;  $y$  axis: the time cost. The labels of the dots indicates the number of transitions occurred before the boat reaches the target:  $\times \rightarrow 0$ ,  $+$   $\rightarrow 1$ ,  $\circ \rightarrow 2$ ,  $\diamond \rightarrow 3$ ,  $*$   $\rightarrow 4$ . Green line: approximate  $u^{r,p}$  (observed average cost); red line:  $u^p$ ; purple line:  $u^r$ . The optimum expected cost  $\approx 0.873$ . The observed averages of coupled (optimal) planning, uncoupled planning and infinite-transition-rate planning are approximately 0.840, 0.882 and 1.030 respectively.

While Figure 4 is based on one particular sequence of switching times, Figures 5 and 6 show the time-of-arrival statistics gathered in 200 Monte-Carlo simulations with  $\lambda = 1$  and  $\lambda = 10$  respectively. In Figure 5 different markers are used to indicate the number of mode switches experienced in each simulation. The green line shows the observed sample average, an approximation of  $u^{r,p}$ , and the red line corresponds to the planner’s naive expectations (based on  $u^0(\mathbf{x}, i)$  and  $u^\infty(\mathbf{x}, i)$  respectively). For reference, the purple line shows the correct attainable minimum average time based on  $u^r(\mathbf{z}) = u^\lambda(\mathbf{z})$ . Figure 6 presents the same information for  $\lambda = 10$ , but uses the  $x$ -axis to represent the number of mode switches. These scatter plots report the arrival times only for those  $\boldsymbol{\alpha}_*^\infty$ -based simulations that reach the target safely.

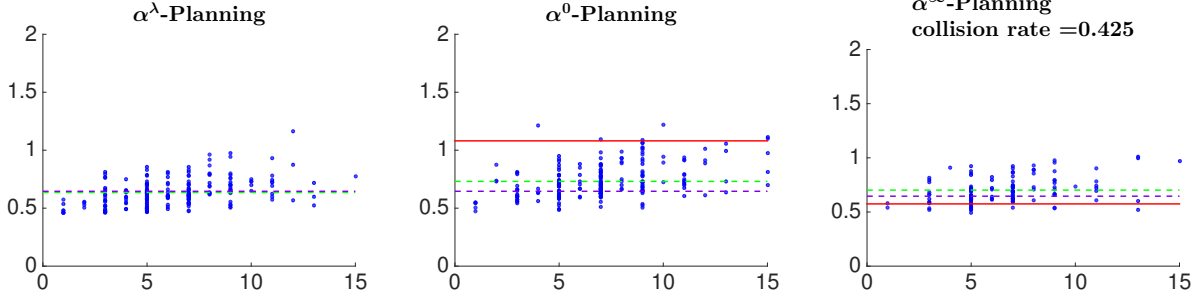


Figure 6: 200 simulations starting at  $((0.5, 0.8), 1)$ , when real transition rate  $\lambda = 10$ .  $x$  axis: the number of transitions that happens in a particular simulation before the swimmer reaches the target;  $y$  axis: the time cost. Green line: approximate  $u^{r,p}$  (observed average cost); red line:  $u^p$ ; purple line:  $u^r$ . The optimum expected cost  $\approx 0.646$ . The observed averages of coupled (optimal) planning, uncoupled planning and infinite-transition-rate planning are approximately 0.636, 0.731 and 0.702 respectively.

We note that for the “coupled planning” our observed sample average is the lowest and quite close to the optimal  $u^\lambda$ . Since  $m(t)$  is based on that  $\lambda$  value, this is hardly surprising. As expected, when  $\lambda$  is low,  $u^\lambda$  is closer to  $u^0$ , and  $u^\infty$  becomes a better approximation for larger  $\lambda$ . But from a practical point of view, a much more relevant question is the relative difference between  $u^{r,p}$  and  $u^r$ . For example,  $(u^{\lambda,0} - u^\lambda)/u^\lambda$  can be used to measure the percentage of performance degradation due to using the uncoupled planning. This quantity is only 1% when  $\lambda = 1$ , but already 13.2% when  $\lambda = 10$ . (The corresponding numbers for the  $\alpha_\infty^*$ -based planning are 18% and 8.7%, but their practical relevance is lower since they are based on a subset of non-colliding trajectories.)

## 6 More Wind Directions and the Limiting Case

### 6.1 Continuous Mode Space Model

In the rowboat problem of section 5, we have assumed that the wind has only two possible directions. However, in real world the wind direction is a continuous variable. We will now consider a different (though also admittedly unrealistic) model, where the wind speed  $s_w$  is constant, the wind velocity is  $\mathbf{w}(\theta) = s_w(\cos \theta, \sin \theta)^T$ , and  $\theta$  undergoes a Brownian Motion on  $\mathbb{R}$ . In this case, it is natural to pose the optimal control problem on an expanded state space  $\tilde{\Omega} = \Omega \times \mathbb{R}$  with the dynamics defined by

$$\begin{cases} \mathbf{y}'(t) &= \mathbf{f}(\mathbf{y}(t), \theta(t), \mathbf{a}(\mathbf{y}(t), \theta(t))), \\ d\theta &= \sigma dW, \\ \mathbf{y}(t) &= \mathbf{x}, \\ \theta(0) &= \theta_0, \end{cases} \quad (6.1)$$

where  $W$  is a standard Wiener process,  $\sigma > 0$ , and  $(\mathbf{x}, \theta_0)$  encode the initial boat position and wind direction. Assuming the running cost  $C(\mathbf{x}, \theta, \alpha)$ , the total cost of this process is

$$J(\mathbf{x}, \theta_0, \alpha) = \int_0^T C(\mathbf{y}(t), \theta(t), \alpha(\mathbf{y}(t), \theta(t))) dt + q(\mathbf{y}(T)), \quad (6.2)$$

and the value function is  $u(\mathbf{x}, \theta) = \inf_{\alpha \in \mathcal{A}} \mathbb{E}[J(\mathbf{x}, \theta, \alpha)]$ . Starting with Bellman’s optimality condition and Taylor expanding (using Ito’s Lemma), we can formally obtain the HJB equation for the

value function:

$$\min_{\mathbf{a} \in A} \{ \nabla_{\mathbf{x}} u(\mathbf{x}, \theta) \cdot \mathbf{f}(\mathbf{x}, \theta, \mathbf{a}) + \frac{\sigma^2}{2} \frac{\partial^2}{\partial \theta^2} u(\mathbf{x}, \theta) + C(\mathbf{x}, \theta, \mathbf{a}) \} = 0. \quad (6.3)$$

Since the wind direction is periodic,  $u(\mathbf{x}, \theta)$  is also periodic with respect to  $\theta$ :

$$u(\mathbf{x}, \theta) = u(\mathbf{x}, \theta + 2\pi), \quad \forall \theta \in \mathbb{R}. \quad (6.4)$$

Therefore, we just need to compute  $u(\mathbf{x}, \theta)$  on  $\Omega \times [0, 2\pi)$ . If we divide  $[0, 2\pi)$  into intervals with length  $\Delta\theta = \frac{2\pi}{n}$ , and introduce discrete values  $\theta_i = i\Delta\theta$  for  $i = 0, 1, \dots, n-1$ , then the second-order term,  $\frac{\sigma^2}{2} \frac{\partial^2}{\partial \theta^2} u(\mathbf{x}, \theta)$ , can be approximated with central difference method. We will use  $\tilde{u}_i(\mathbf{x}) = \tilde{u}(\mathbf{x}, \theta_i)$  to denote this  $\theta$ -discretized approximation of  $u(\mathbf{x}, i\Delta\theta)$ . These functions must satisfy the system of first-order HJB PDEs

$$\min_{\mathbf{a} \in A} \{ \nabla_{\mathbf{x}} \tilde{u}_i(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x}, \theta_i, \mathbf{a}) + C(\mathbf{x}, \theta_i, \mathbf{a}) \} + \frac{\sigma^2}{2(\Delta\theta)^2} \sum_{\epsilon=\pm 1} (\tilde{u}_{i+\epsilon}(\mathbf{x}) - \tilde{u}_i(\mathbf{x})) = 0; \quad i = 0, 1, \dots, n-1, \quad (6.5)$$

where we have used the notation  $\tilde{u}_n(\mathbf{x}) = \tilde{u}_0(\mathbf{x})$  and  $\tilde{u}_{-1}(\mathbf{x}) = \tilde{u}_{n-1}(\mathbf{x})$ .

This can be also interpreted as a piecewise-deterministic system discussed in section 2, provided

- (a) from every mode  $i = 0, \dots, (n-1)$ , the switching can happen only to its “adjacent” modes  $(i-1)$  and  $(i+1)$ ;
- (b) the rate of switching to these adjacent modes is equal to  $\frac{\sigma^2}{2(\Delta\theta)^2} = \frac{\sigma^2 n^2}{8\pi^2}$ .

As the number of modes grows, the original value function on  $\mathbb{R}^{d+1}$  is recovered in the limit:

$$u(\mathbf{x}, \theta) = \lim_{n \rightarrow \infty} \tilde{u}\left(\mathbf{x}, \frac{2\pi}{n} \left\lfloor \frac{n\theta}{2\pi} \right\rfloor\right).$$

## 6.2 Upper Bound on $\theta$ -variability

It is easy to derive a uniform upper bound on  $|u(\mathbf{x}, \theta_1) - u(\mathbf{x}, \theta_2)|$ ,  $\forall \mathbf{x}, \theta_1, \theta_2$ . Suppose the *first hitting time* is  $\kappa(\theta_1, \theta_2) = \min_{t \geq 0} \{t \mid \theta(t_0 + t) = \theta_2 + 2n\pi \text{ for some integer } n \mid \theta(t_0) = \theta_1\}$ . Since  $u$  is periodic in  $\theta$ , we just need to consider the case  $\theta_2 - 2\pi < \theta_1 < \theta_2 < \theta_1 + 2\pi$  and  $\mathbb{E}[\kappa(\theta_2, \theta_1)] = \phi(\theta_2 - \theta_1)$ , where  $\phi(\alpha)$  is the solution of

$$\phi''(\alpha) = -\frac{2}{\sigma^2}, \quad \forall \alpha \in (0, 2\pi); \quad \phi(0) = \phi(2\pi) = 0.$$

Thus,  $\phi(\alpha) = \alpha(2\pi - \alpha)/\sigma^2 \leq \frac{\pi^2}{\sigma^2}$ . By symmetry the same formula is also valid for  $\mathbb{E}[\kappa(\theta_1, \theta_2)]$ . An argument similar to that in Appendix A, shows that

$$|u(\mathbf{x}, \theta_1) - u(\mathbf{x}, \theta_2)| \leq \max\{\mathbb{E}[\kappa(\theta_1, \theta_2)], \mathbb{E}[\kappa(\theta_2, \theta_1)]\} \leq -\frac{1}{\sigma^2}(\theta_2 - \theta_1)(\theta_2 - \theta_1 - 2\pi) \leq \frac{\pi^2}{\sigma^2}.$$

Similar upper bounds are also easy to derive for  $\|\tilde{u}_i - \tilde{u}_j\|_\infty$  in the  $\theta$ -discretized version.

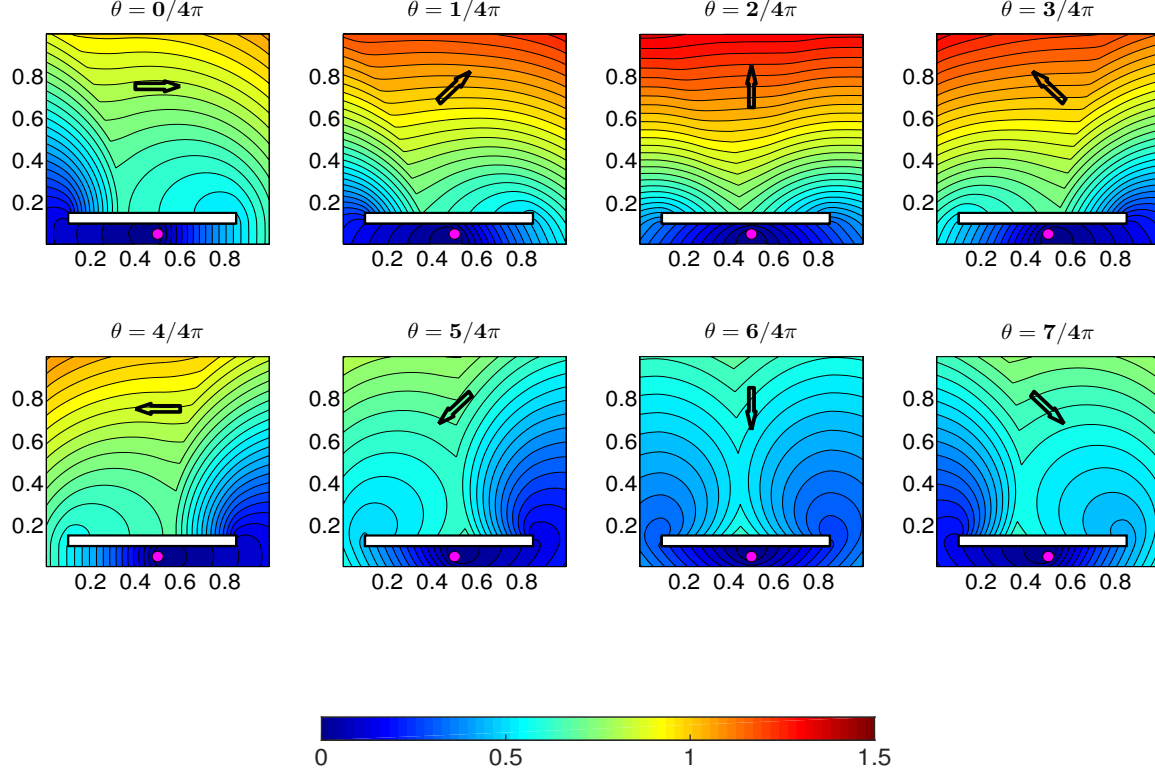


Figure 7: Value functions for 8 wind modes discretizing the degenerate diffusion model in section 6. Wind directions represented by arrows.

### 6.3 Example

Returning to the problem of time-optimal trajectories (i.e.,  $C \equiv 1$ ), we will assume that the rowing speed is  $s(\mathbf{x}) = 2$ , while the wind speed is  $s_w = 1.5$  and the wind direction undergoes Brownian motion with  $\sigma = 2$ . The resulting second-order HJB PDE is

$$2\|\nabla u\| - \mathbf{w}(\theta) \cdot \nabla u = 1 + 2u_{\theta\theta},$$

where  $\mathbf{w}(\theta) = 1.5(\cos(\theta), \sin(\theta))^T$ . To treat this in the framework of section 5, we discretize in  $\theta$  using  $n$  different wind modes with

$$\theta_i = \frac{2i\pi}{n}; \quad \mathbf{w}_i = \mathbf{w}(\theta_i); \quad \lambda_{ij} = \begin{cases} \frac{n^2}{2\pi^2}, & j \equiv i \pm 1 \pmod{n}; \\ 0, & \text{otherwise;} \end{cases} \quad i, j = 0, \dots, n-1.$$

Figure 7 presents the numerical results for  $n = 8$  wind modes.

## 7 Conclusions

We have considered the problem of optimal path planning under uncertainty due to stochastic switching of *modes* in the system dynamics. Such piecewise-deterministic models result in weakly-coupled systems of static HJB equations, and we have discussed both the discretization strategies and the iterative numerical algorithms for solving them. Asymptotic approximations, obtained by letting the

rates of switching tend to either zero or  $\infty$ , are less challenging computationally. But our experimental evidence showed that the use of controls recovered from them results in significant performance degradation in non-asymptotic regimes. The model problems used in our computational experiments are quite simple, but already reveal the complications and opportunities of taking stochastic switching into account. Some generalizations (e.g., more modes, general anisotropic cost/dynamics, or spatially-dependent switching rates) would be trivial. It would be both more challenging and interesting to incorporate some non-Markovian switching.

From practitioner’s perspective, piecewise-deterministic models will become much more attractive once efficient numerical methods become available. It will be interesting to see if the number of iterations can be significantly reduced despite the mode-coupling and the mode-dependence of optimal directions. This is obviously doable for a subclass of *structurally causal* switching systems: if the modes can be re-numbered so that the transition matrix  $\Lambda = (\lambda_{ij})$  becomes upper triangular, the equations in (2.9) can be de-coupled and solved one at a time. Examples of this type are common in some of the applications (e.g., [28, 27, 3]), but it is still unclear whether similar efficiency gains are attainable in the general case. Another relevant practical task is to derive a priori upper bounds on performance degradation due to using  $\alpha_*^0$  or  $\alpha_*^\infty$  instead of the correct  $\alpha_*$ . Practitioners could use this information to decide whether solving the coupled system is actually worthwhile.

Many features already available for the fully deterministic case would be similarly attractive in the piecewise-deterministic setting. This includes multi-criteria path planning [31] and dynamic domain restriction [16]. It will be similarly useful to extend the concept of *risk-sensitive* optimal controls, already popular in standard stochastic control problems [25]. Another alternative is to optimize the expected cost with a constraint on the “worst case scenario” [21], since the latter is well-defined in stochastic switching problems.

## A Upper Bound for Mode Differences

Our goal is to derive an upper bound on the difference between mode-specific value functions:  $\|u_i - u_j\|_\infty$  with  $i, j \in \mathcal{M}$ . The basic idea is that, starting from  $\mathbf{x}$  in the mode  $i$ , it is often possible to hover around  $x$  long enough until we transition into a (possibly preferable) mode  $j$ . We define *the first hitting time*  $\kappa_{ij} = \min_{t \geq 0} \{t : m(t_0 + t) = j \mid m(t_0) = i\}$  with  $\kappa_{ii} = 0$ ,  $\forall i \in \mathcal{M}$ , and make the following assumptions about the controlled process.

**Assumption A.1.** *The running cost  $C_i(\mathbf{x}, \mathbf{a}) \leq \bar{C}$  for all  $(\mathbf{x}, i, \mathbf{a}) \in \Omega \times \mathcal{M} \times A$ .*

**Assumption A.2.**  *$u_i(\mathbf{x})$  is  $L$ -Lipschitz continuous for all  $i \in \mathcal{M}$ . That is,*

$$|u_i(\mathbf{x}_1) - u_i(\mathbf{x}_2)| \leq L \|\mathbf{x}_1 - \mathbf{x}_2\|, \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \Omega, i \in \mathcal{M} \quad (\text{A.1})$$

**Assumption A.3.** *For every starting configurations  $(\mathbf{x}, i) \in \Omega \times \mathcal{M}$  and every  $\delta > 0$ , there exists a control  $\alpha_\delta \in \mathcal{A}$  such that*

$$\|\mathbf{y}(t) - \mathbf{x}\| < \delta, \quad \forall t > 0, \quad (\text{A.2})$$

where  $\mathbf{y}$  satisfies equation (2.1) with  $\alpha = \alpha_\delta$ .

**Theorem A.1.** *If the assumptions A.1-A.3 hold, then*

$$|u_i(\mathbf{x}) - u_j(\mathbf{x})| \leq \bar{C} \max\{\mathbb{E}[\kappa_{ij}], \mathbb{E}[\kappa_{ji}]\}, \quad \forall i, j \in \mathcal{M}. \quad (\text{A.3})$$

*Proof.* Starting from  $\mathbf{x} \in \Omega$  in mode  $i$ , we choose a number  $\delta > 0$  small enough such that  $\delta < \text{dist}(\mathbf{x}, Q)$ , and introduce the following hybrid strategy (denoted  $\tilde{\alpha}_\delta$ ). We use  $\alpha_\delta(\mathbf{x}, m(t))$  described

in assumption A.3 until the mode switches to  $j$  at some point  $\tilde{\mathbf{x}} = \mathbf{y}(\kappa_{ij})$ . From there on, we switch to the optimal strategy corresponding to  $(\tilde{\mathbf{x}}, j)$ .

While  $t \leq \kappa_{ij}$ , we have  $\|\mathbf{y}(t) - \mathbf{x}\| < \delta < \text{dist}(\mathbf{x}, Q)$ ; so the trajectory will not reach  $Q$  before the mode switches to  $j$ . Therefore,

$$u_i(\mathbf{x}) \leq \mathbb{E}[J(\mathbf{x}, i, \tilde{\alpha}_\delta)] \leq \bar{C} \mathbb{E}[\kappa_{ji}] + \mathbb{E}[J(\tilde{\mathbf{x}}, j, \tilde{\alpha}_\delta)] = \bar{C} \mathbb{E}[\kappa_{ij}] + u_j(\tilde{\mathbf{x}}). \quad (\text{A.4})$$

Since  $\|\tilde{\mathbf{x}} - \mathbf{x}\| < \delta$ , inequalities (A.1) and (A.4) yield

$$u_i(\mathbf{x}) \leq \bar{C} \mathbb{E}[\kappa_{ij}] + u_j(\mathbf{x}) + L\delta. \quad (\text{A.5})$$

Reversing the roles of  $i$  and  $j$ , we also have

$$u_j(\mathbf{x}) \leq \bar{C} \mathbb{E}[\kappa_{ji}] + u_i(\mathbf{x}) + L\delta. \quad (\text{A.6})$$

To finish the proof, we combine (A.5) and (A.6), and then let  $\delta \rightarrow 0^+$ .  $\square$

Suppose the transition rate matrix is  $c\Lambda$  for  $c > 0$  and a fixed irreducible matrix  $\Lambda$ . If we define  $\rho_{ij} = \max\{\mathbb{E}[\kappa_{ij}], \mathbb{E}[\kappa_{ji}]\}$ , a standard argument from the theory of Markov processes [33] shows that  $\rho_{ij} \propto c^{-1}$ . Therefore, if we send  $c \rightarrow \infty$ ,  $|u_i(\mathbf{x}) - u_j(\mathbf{x})| = O(c^{-1})$  uniformly in  $\Omega$  for all  $i \neq j$ .

#### \*Bibliography

- [1] Ramakrishna Akella and P Roshan Kumar. Optimal control of production rate in a failure prone manufacturing system. *Automatic Control, IEEE Transactions on*, 31(2):116–126, 1986.
- [2] Ken Alton and Ian M Mitchell. An ordered upwind method with precomputed stencil and monotone node acceptance for solving static convex hamilton-jacobi equations. *Journal of Scientific Computing*, 51(2):313–348, 2012.
- [3] June Andrews and Alexander Vladimirovsky. Deterministic control of randomly-terminated processes. *Interfaces and Free Boundaries*, 16(1):1–40, 2014.
- [4] Stanley Bak, Joyce McLaughlin, and Daniel Renzi. Some improvements for the fast sweeping method. *SIAM Journal on Scientific Computing*, 32(5):2853–2874, 2010.
- [5] Martino Bardi and Italo Capuzzo-Dolcetta. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Springer Science & Business Media, 2008.
- [6] Guy Barles and Panagiotis E Souganidis. Convergence of approximation schemes for fully nonlinear second order equations. In *Asymptotic analysis*, number 3, pages 271–283, 1991.
- [7] E Barron, L Evans, and R Jensen. The infinity laplacian, aronsson’s equation and their generalizations. *Transactions of the American Mathematical Society*, 360(1):77–101, 2008.
- [8] Alain Bensoussan and Jose Luis Menaldi. Stochastic hybrid control. *Journal of Mathematical Analysis and Applications*, 249(1):261–288, 2000.
- [9] T Bielecki and PR Kumar. Optimality of zero-inventory policies for unreliable manufacturing systems. *Operations research*, 36(4):532–541, 1988.
- [10] Folkmar Bornemann and Christian Rasch. Finite-element discretization of static hamilton-jacobi equations based on a local variational principle. *Computing and Visualization in Science*, 9(2):57–69, 2006.
- [11] Michelle Boué and Paul Dupuis. Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control. *SIAM Journal on Numerical Analysis*, 36(3):667–695, 1999.
- [12] El Kébir Boukas, Alain Haurie, and Philippe Michel. An optimal control problem with a random stopping time. *Journal of optimization theory and applications*, 64(3):471–480, 1990.
- [13] Manuela L Bujorianu and John Lygeros. General stochastic hybrid systems: Modelling and optimal control. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 2, pages 1872–1877. IEEE, 2004.

- [14] Adam Chacon and Alexander Vladimirsky. Fast two-scale methods for eikonal equations. *SIAM Journal on Scientific Computing*, 34(2):A547–A578, 2012.
- [15] Adam Chacon and Alexander Vladimirsky. A parallel two-scale method for eikonal equations. *SIAM Journal on Scientific Computing*, 37(1):A156–A180, 2015.
- [16] Z. Clawson, A. Chacon, and A. Vladimirsky. Causal domain restriction for eikonal equations. *SIAM Journal on Scientific Computing*, 36(5):A2478–A2505, 2014.
- [17] Michael G Crandall and Pierre-Louis Lions. Viscosity solutions of hamilton-jacobi equations. Technical report, DTIC Document, 1981.
- [18] Emiliano Cristiani, Fabio S Priuli, and Andrea Tosin. Modeling rationality to control self-organization of crowds: an environmental approach. *SIAM Journal on Applied Mathematics*, 75(2):605–629, 2015.
- [19] D Dahiya, S Baskar, and F Coulouvrat. Characteristic fast marching method for monotonically propagating fronts in a moving medium. *SIAM Journal on Scientific Computing*, 35(4):A1880–A1902, 2013.
- [20] MHA Davis. Control of piecewise-deterministic processes via discrete-time dynamic programming. In *Stochastic Differential Systems*, pages 140–150. Springer, 1986.
- [21] Stefano Ermon, Carla P. Gomes, Bart Selman, and Alexander Vladimirsky. Probabilistic planning with non-linear utility functions and worst-case guarantees. In Wiebe van der Hoek, Lin Padgham, Vincent Conitzer, and Michael Winikoff, editors, *AAMAS*, pages 965–972. IFAA-MAS, 2012.
- [22] Lawrence C Evans. The perturbed test function method for viscosity solutions of nonlinear pde. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics*, 111(3-4):359–375, 1989.
- [23] M Falcone. The minimum time problem and its applications to front propagation. *Motion by mean curvature and related topics (Trento, 1992)*, de Gruyter, Berlin, pages 70–88, 1994.
- [24] Maurizio Falcone and Roberto Ferretti. *Semi-Lagrangian approximation schemes for linear and Hamilton-Jacobi equations*. SIAM, 2013.
- [25] Wendell H Fleming and Halil Mete Soner. *Controlled Markov processes and viscosity solutions*, volume 25. Springer Science & Business Media, 2006.
- [26] Tor Gillberg, Mohammed Sourouri, and Xing Cai. A new parallel 3d front propagation algorithm for fast simulation of geological folds. *Procedia Computer Science*, 9:947–955, 2012.
- [27] Alain Haurie. A multigenerational game model to analyze sustainable development. *Annals of Operations Research*, 137(1):369–386, 2005.
- [28] Alain Haurie and Francesco Moresino. A stochastic control model of economic growth with environmental disaster prevention. *Automatica*, 42(8):1417–1428, 2006.
- [29] Won-Ki Jeong, P Thomas Fletcher, Ran Tao, and Ross T Whitaker. Interactive visualization of volumetric white matter connectivity in dt-mri using a parallel-hardware hamilton-jacobi solver. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1480–1487, 2007.
- [30] Chiu-Yen Kao, Stanley Osher, and Yen-Hsi Tsai. Fast sweeping methods for static hamilton-jacobi equations. *SIAM journal on numerical analysis*, 42(6):2612–2632, 2005.
- [31] A. Kumar and A. Vladimirsky. An efficient method for multiobjective optimal control and optimal control subject to integral constraints. *Journal of Computational Mathematics*, 28(4):517–551, 2010.
- [32] Jean-Marie Mirebeau. Efficient fast marching with finsler metrics. *Numerische Mathematik*, 126(3):515–557, 2014.
- [33] James R Norris. *Markov chains*. Number 2008. Cambridge University Press, 1998.
- [34] GJ Olsder and R Suri. Time-optimal control of parts-routing in a manufacturing system with failure-prone machines. In *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, pages 722–727. IEEE, 1980.

- [35] Jianliang Qian, Yong-Tao Zhang, and Hong-Kai Zhao. A fast sweeping method for static convex hamilton–jacobi equations. *Journal of Scientific Computing*, 31(1):237–271, 2007.
- [36] Suresh P Sethi and Qing Zhang. *Hierarchical decision making in stochastic manufacturing systems*. Springer Science & Business Media, 2012.
- [37] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.
- [38] James A Sethian and Alexander Vladimirsky. Ordered upwind methods for static hamilton–jacobi equations. *Proceedings of the National Academy of Sciences*, 98(20):11069–11074, 2001.
- [39] James A Sethian and Alexander Vladimirsky. Ordered upwind methods for static hamilton–jacobi equations: Theory and algorithms. *SIAM Journal on Numerical Analysis*, 41(1):325–363, 2003.
- [40] D Verms. Optimal control of piecewise deterministic markov process. *Stochastics: An International Journal of Probability and Stochastic Processes*, 14(3):165–207, 1985.
- [41] Dmitry S Yershov and Steven M LaValle. Simplicial label correcting algorithms for continuous stochastic shortest path problems. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5062–5067. IEEE, 2013.
- [42] Hongkai Zhao. A fast sweeping method for eikonal equations. *Mathematics of computation*, 74(250):603–627, 2005.
- [43] Sanjo Zlobec. Zermelo’s navigation problems. In *Stable Parametric Programming*, pages 213–223. Springer, 2001.